## WHAT IS CLAIMED IS:

- 1 1. A processor comprising:
- an out-of-order microinstruction pointer (μIP) stack in a
- 3 microcode (µcode) execution core.
- 1 2. The processor of claim 1 in which the  $\mu$ IP stack
- comprises:
- an entry number field;
- a microinstruction pointer (μΙΡ) field;
- 5 a back pointer field;
- a retirement indicator field; and
- 7 a return pointer field.
- 1 3. The processor of claim 2 in which the  $\mu$ IP field is
- 2 14-bits wide.
- 1 4. The processor of claim 3 in which the  $\mu$ IP field has
- a microinstruction pointer ( $\mu$ IP) pushed by a first
- microoperation ( $\mu$ Op) code and used by a second  $\mu$ Op code.
- 1 5. The processor of claim 2 in which the back pointer
- 2 field has a pointer to a next entry in the  $\mu$ IP stack for a
- 3 micro-type of service ( $\mu TOS$ ) bit to point to after a  $\mu Op$ .

- 1 6. The processor of claim 2 in which the retirement
- 2 indicator field has an indication of whether an entry has
- 3 retired.
- The processor of claim 2 in the return pointer field
- a pointer to a location in a retirement stack to which an
- 3 entry is copied after being retired.
- 1 8. A method executed in a processor comprising:
- 2 executing microcode (μcode) stored in an out-of-
- order microinstruction pointer (μΙΡ) stack; and
- manipulating the  $\mu$ IP stack with a set of
- 5 microinstructions.
- 1 9. The method of claim 8 in which the stack has an
- entry number field, a microinstruction pointer (µIP) field, a
- 3 back pointer field, a retirement indicator field and a return
- 4 pointer field.
- 1 10. The method of claim 9 in which the  $\mu$ IP pointer field
- 2 is 14-bits wide.
- 1 11. The method of claim 10 in which the  $\mu$ IP pointer
- field has a microinstruction pointer (µIP) pushed by a first
- 3 microoperation ( $\mu$ Op) code and used by a second  $\mu$ Op code.

- 1 12. The method of claim 9 in which the back pointer
- 2 field has a pointer to a next entry in the µIP stack for a
- 3 micro-type of service ( $\mu TOS$ ) bit to point to after a  $\mu Op$ .
- 1 13. The method of claim 9 in which the retirement
- 2 indicator field has an indication of whether an entry has
- 3 retired.
- 1 14. The method of claim 9 in which the return pointer
- field contains a pointer to a location in a retirement stack
- 3 to which an entry is copied after being retired.
- 1 15. The method of claim 9 in which manipulating
- 2 comprises:
- pushing a next μIP on to the μIP stack; and
- using the next  $\mu$ IP in an intermediate field as a target
- 5  $\mu$ IP in a jump operation.
- 1 16. The method of claim 9 in which manipulating
- 2 comprises:
- 3 taking a value of an intermediate field of a
- 4 microoperation (μOp); and
- 5 pushing the value on to the  $\mu$ IP stack.

1

- 1 17. The method of claim 9 in which manipulating
- comprises:
- popping a value off the μIP stack; and
- 4 replacing a current μOp intermediate field.
- 1 18. The method of claim 9 in which manipulating
- comprises:
- popping a value off of the μIP stack; and
- 4 jumping to that value.
- 1 19. The method of claim 9 in which manipulating
- comprises:
- reading a value off the μIP stack; and
- 4 replacing a  $\mu$ Op's intermediate field with the value.
- 1 20. The method of claim 9 in which manipulating
- comprises setting the  $\mu$ IP stack pointers to reset.
- 1 21. The method of claim 9 further comprising providing a
- set of pointers that point to different entries in the  $\mu$ IP
- 3 stack.
- 1 22. The method of claim 21 in which the set of pointers
- 2 includes a  $\mu TOS$  pointer that points to a top of the  $\mu IP$  stack.

- 1 23. The method of claim 21 in which the set of pointers
- 2 includes a  $\mu$ Alloc pointer that points to a next allocated
- 3 entry in the μIP stack.
- 1 24. The method of claim 21 in which the set of pointers
- 2 includes a NextRet pointer that points to a next entry in the
- $\mu$ IP stack to be deallocated.
- 1 25. The method of claim 21 in which the set of pointers
- 2 includes  $\mu Ret Tos$  pointer that points at a retired top of the
- 3 μIP stack.
- 1 26. The method of claim 8 in which the  $\mu$ OPs include an
- 2 ms\_call  $\mu$ OP that takes a next  $\mu$ IP, pushes the next  $\mu$ IP on the
- 3  $\mu$ IP stack, and uses the next  $\mu$ IP in an intermediate field as a
- 4 target μIP of a jump.
- 1 27. The method of claim 8 in which the  $\mu$ OPs include an
- 2 ms\_push µOP that takes a value in an intermediate field and
- 3 pushes the value on the  $\mu$ IP stack.
- 1 28. The method of claim 8 in which the  $\mu OPs$  include an
- 2 ms pop  $\mu$ OP that pops a value off the  $\mu$ IP stack and replaces
- 3 the value with the  $\mu OP$ 's intermediate field.

- 1 29. The method of claim 8 in which the  $\mu$ OPs include an
- 2 ms return  $\mu$ OP that pops a value off of the  $\mu$ IP stack and jumps
- 3 to that  $\mu$ IP.
- 1 30. The method of claim 8 in which the  $\mu$ OPs include an
- 2 ms tos read  $\mu$ OP that reads a value off the  $\mu$ IP stack and
- 3 replaces this  $\mu$ OP's intermediate field.
- 1 31. The method of claim 8 in which the  $\mu$ OPs include an
- 2 ms\_ $\mu$ ip\_stack\_clear  $\mu$ OP that sets the  $\mu$ IP stack pointers to
- 3 reset.
- 1 32. A computer program product residing on a computer
- 2 readable medium having instructions stored thereon which, when
- 3 executed by the processor, cause the processor to:
- execute microcode (µcode) stored in an out-of-order
- 5 microinstruction pointer (μIP) stack; and
- manipulate the  $\mu$ IP stack with a set of microinstructions.
- 1 33. The computer program product of claim 32 wherein
- 2 instructions to manipulate further comprise instructions to:
- g push a next  $\mu$ IP on to the  $\mu$ IP stack; and
- 4 use the next  $\mu$ IP in an intermediate field as a target  $\mu$ IP
- 5 in a jump operation.

1

2

- 1 34. The computer program product of claim 32 wherein
  2 instructions to manipulate further comprise instructions to:
  3 take a value of an intermediate field of a microoperation
  4 (μOp); and
  push the value on to the μIP stack.
- 35. The computer program product of claim 32 wherein
   instructions to manipulate further comprise instructions to:
   pop a value off the μIP stack; and
   replace a current μOp intermediate field with the value.
  - 36. The computer program product of claim 32 wherein instructions to manipulate further comprise instructions to: pop a value off of the  $\mu$ IP stack; and jump to that value.
- 1 37. The computer program product of claim 32 wherein 2 instructions to manipulate further comprise instructions to: 3 read a value off the  $\mu$ IP stack; and 4 replace a  $\mu$ Op's intermediate field with the value.
- 1 38. The computer program product of claim 32 wherein
  2 instructions to manipulate further comprise instructions to:
  3 set the µIP stack pointers to reset.